Fast parallel algorithm for three-dimensional distance-driven model in iterative computed tomography reconstruction*

Chen Jian-Lin(陈建林), Li Lei(李 磊), Wang Lin-Yuan(王林元), Cai Ai-Long(蔡爱龙), Xi Xiao-Qi(席晓琦), Zhang Han-Ming(张瀚铭), Li Jian-Xin(李建新), Yan Bin(闫 镔)†

National Digital Switching System Engineering & Technological R & D Center, Zhengzhou 450002, *China*

(Received 25 June 2014; revised manuscript received 11 September 2014; published online 10 December 2014)

The projection matrix model is used to describe the physical relationship between reconstructed object and projection. Such a model has a strong influence on projection and backprojection, two vital operations in iterative computed tomographic reconstruction. The distance-driven model (DDM) is a state-of-the-art technology that simulates forward and back projections. This model has a low computational complexity and a relatively high spatial resolution; however, it includes only a few methods in a parallel operation with a matched model scheme. This study introduces a fast and parallelizable algorithm to improve the traditional DDM for computing the parallel projection and backprojection operations. Our proposed model has been implemented on a GPU (graphic processing unit) platform and has achieved satisfactory computational efficiency with no approximation. The runtime for the projection and backprojection operations with our model is approximately 4.5 s and 10.5 s per loop, respectively, with an image size of 256×256×256 and 360 projections with a size of 512×512. We compare several general algorithms that have been proposed for maximizing GPU efficiency by using the unmatched projection/backprojection models in a parallel computation. The imaging resolution is not sacrificed and remains accurate during computed tomographic reconstruction.

Keywords: computed tomography, iterative reconstruction, parallelizable algorithm, distance-driven model

PACS: 87.59.–e, 07.85.–m, 87.57.Q– **DOI:** 10.1088/1674-1056/24/2/028703

1. Introduction

Computed tomographic (CT) reconstructions of three-dimensional (3D) objects reconstructed from a set of integrals from projection data are becoming increasingly popular. CT has the advantages of high efficiency, fast scanning speed, flexible scanning scope, and high image resolution. Iterative reconstruction methods[1–6] have attracted considerable interest in recent years because they can cope well with limited projection sets and noisy data. However, these methods demand huge computational cost.[1,2] In iterative reconstruction, projection and backprojection as two vital operations are repeated per iteration and are time-consuming in reconstruction processing.

In iterative reconstruction algorithms, the projection and backprojection procedures for a discrete imaging object have been modeled. Different types of system matrix models have been studied for weighing the relationship between discrete projection data and discrete reconstructed object. These models compromise computational complexity and accuracy. Numerous approaches have been proposed to express system matrix coefficients. Conventional models can be classified into the following categories:[7] intersection length model,[8] linear interpolation model,[9,10] distance–distance model,[11] basis function model,[12] finite beam width model,[13] and Fourier-based projection model.[14] Simple models, such as intersection length and linear interpolation models, are often applied in computed iterative reconstruction because of their computing convenience. In addition, these models can be used for high-frequency artifacts. Exact models, such as basic function and finite beam width models, can hardly be used for computed iterative tomography, particularly in 3D reconstruction, because they suffer from computational inefficiency.

For implementing the projection and backprojection operations in 3D reconstruction, the corresponding sparse matrix-vector multiplication (SpMV)[15] method based on compressing matrices can be used given that the system matrix is sparse. However, SpMV only adapts to a limited scale. The calculation of the system matrix coefficients usually needs to be repeated and the computing effect usually needs to be sacrificed to change for the need of hardware source. In recent years, high-performance computation using programmable commodity hardware has been a useful tool in medical imaging. Particularly, GPUs offer an attractive alternative platform to cope with the demands in CT for the high peak performance, cost effectiveness, and availability of a user-friendly programming environment.[15] The methods for realizing the projection and backprojection operations are usually classified into two categories, namely, ray-driven[8] and voxel-driven methods.[16] In the voxel-driven method, the detectors are updated with voxel values using similar weights in the projection

*Projected supported by the National High Technology Research and Development Program of China (Grant No. 2012AA011603) and the National Natural Science Foundation of China (Grant No. 61372172).

†Corresponding author. E-mail: tom.yan@gmail.com

© 2015 Chinese Physical Society and IOP Publishing Ltd

process. In the ray-driven method, the voxels are updated with detector values. However, in the reconstruction implementation, unmatched projector–back projector pairs are often used for GPU-accelerated methods to cater for the approximation need and to maximize the efficiency of GPUs. These projectors include ray-driven and pixel-driven back projectors. It can cause additional artifacts in reconstruction, which propagate through iterations and decrease reconstruction accuracy, particularly when the number of iterations is increased.[17]

This study introduces a new fast parallel alogrithm for a 3D distance-driven model (DDM) in cone-beam CT (CBCT) reconstruction. The method has the advantage of the distance-driven model with a low arithmetic cost and good approximation to the true strip integral compared with general models, such as the intersection length model or linear interpolation model. The method can be fit for parallel computing and either the projection onto a given detector cell or the back-projection onto a given voxel has been calculated and updated per independent thread on a GPU platform. This method significantly improves the computational efficiency, and the result is similar to the non-accelerated model. Our method provides a matched projector–backprojector pair, and the system matrix of projection is the adjoint of backprojection matrix. Compared with other GPU-accelerated methods with unmatched projector–backprojector pair, our method provides a more accurate performance in image reconstruction.

2. Method

2.1. Discrete–discrete imaging model

CT scanners can be modeled using several imaging systems by the following linear equation

$$W\boldsymbol{f} = \boldsymbol{p}, \tag{1}$$

where $\boldsymbol{p} \in \boldsymbol{P}$ represents projection data, $\boldsymbol{f} \in \boldsymbol{F}$ represents a pre-reconstructed image, and the system matrix $W : \boldsymbol{F} \to \boldsymbol{P}$ is a projection operator. In practice, the discrete–discrete model is assumed. For the 3D case, an image f can be discretized by adding a cube grid on the image, and the projection data p are usually measured with detector cells. As a result, we develop a 3D image $f = (f_{i,j,k}) \in R^{Nx} \times R^{Ny} \times R^{Nz}$, where the indices $1 \le i \le N_x$, $1 \le j \le N_y$, $1 \le k \le N_z$ are integers. The geometry of the x-ray imaging system in CBCT can be illustrated as shown in Fig. 1.

In CBCT, equation (1) can be rewritten as

$$p_{m,\theta} = \sum_{n=1}^{N} W_{m\theta,ijk} f_{i,j,k}, \quad m = 1,2,\ldots,M, \tag{2}$$

where $f_{i,j,k}$ is the underlying source voxel, $p_{m,\theta}$ is the projection measured in the m-th finite-width detector cell at angle θ, and $W_{m\theta,ijk} \ge 0$ denotes the element of the system matrix. Equation (2) can also be regarded as the general expression of the projection. The system coefficient $W_{m\theta,ijk}$ weighs the contribution of voxel $f_{i,j,k}$ to the m-th detector cell at angle θ. The discrete–discrete model of backprojection operation can be homoplastically expressed by

$$f_{i,j,k} = \sum_{m,\theta} W_{m\theta,ijk} p_{m,\theta}. \tag{3}$$

In this study, we develop parallel computing methods that can rapidly compute the projection and backprojection operations on the fly.

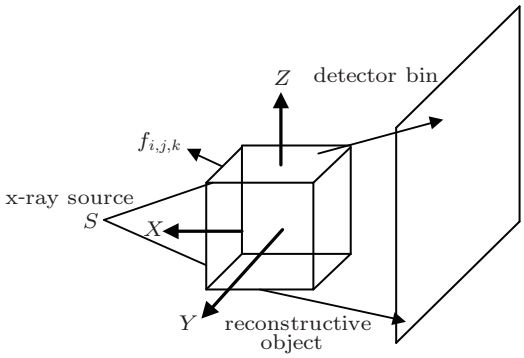


Fig. 1. The geometry of the x-ray imaging system. S represents the x-ray source, and $f_{i,j,k}$ represents the 3D reconstructed object.

2.2. Parallelizing 3D distance-driven forward projection and backprojection

In two-dimensional (2D) DDMs, both pixel and detector boundaries are mapped onto a common axis, and overlapped areas can be normalized and approximated as the system matrix coefficient (Fig. 2). The 2D image pixels are f_i $(i = 1,2,3)$ and the detector bins are p_i $(i = 1,2,3)$. Both the pixel boundaries and the detector bin boundaries are mapped on a common axis and the coordinates of pixel projection points are x_n, x_{n+1}, x_{n+2}, x_{n+3} while the y_m, y_{m+1}, y_{m+2}, y_{m+3} represent the coordinates of detector bins projection points. For example, the forward projection value of the detector bin p_2 can be calculated as

$$\Delta_{p2} = \frac{(x_{n+1} - y_{m+1}) f_1 + (y_{m+2} - x_{n+1}) f_2}{x_{n+2} - y_{m+2}}, \tag{4}$$

and the backprojection value of the pixel f_2 can be shown as

$$\Delta f_2 = \frac{(y_{m+2} - x_{n+1}) p_2 + (x_{n+2} - y_{m+2}) p_3}{x_{n+2} - x_{n+1}}, \tag{5}$$

Nguyen *et al.*[18] designed a novel scheme to solve the 2D DDM for parallel computation. In the study, the forward projection and backprojection operations can be realized on GPUs. These operations do not cause "writing conflict." The key for the algorithm is to utilize the detector and pixel boundaries to find a set of pixels and detector bins, respectively. They also explained that their method can be extended to the improved distance-driven method called the separate footprint method.[19]

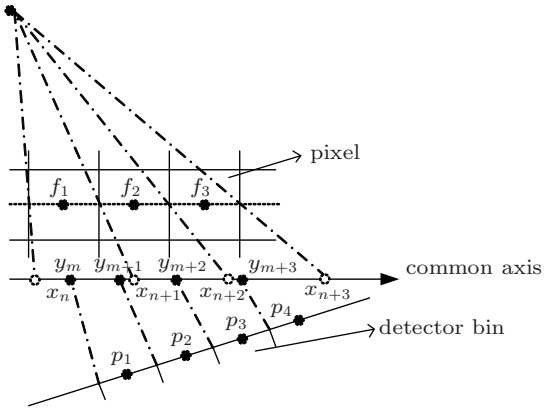


Fig. 2. The 2D distance-driven model.

De Man and Basu[20] introduced the 3D method. In the 3D DDM, the horizontal and vertical boundaries of the image voxels and detector cells are mapped onto a common plane, and the overlap lengths along the horizontal and vertical directions are calculated, respectively. Then, the overlap area is obtained by multiplying the two lengths. This method approximates the shape of a rectangle and simplifies the calculation.

For the traditional sequential processing for both projection and backprojection, the efficient method is to map all the voxel boundaries and the detector cells boundaries in a 3D image slice onto a common plane along the horizontal and vertical directions separately and then to sequentially compute the two normalized overlap lengths between the voxels and detector bins by using the intersections on the common plane. Whereas, this scheme is not optimal for parallel computing, because computing the normalized length each time along the common plane updates many detector bins in projection and many voxels in backprojection. In this study, we proposed a fast parallel algorithm to solve the parallel computing for the 3D distance-model in the forward projection and backprojection.

2.2.1. Forward projection

In this study, we consider a 3D image that consists of (N_x, N_y, N_z) voxels with a size of (d_x, d_y, d_z). Let $N_x = N_y$, $d_x = d_y = d_z = 1$ and the center point of the image at the origin is defined. We also define that $f_{i,j,k}$ represents the voxels of the reconstructed object and (x, y, z) is the location coordinates, and i, j, k are the voxel indices.

Moreover, we define that the detector consists of (N_s, N_t) pixels with a size of (d_s, d_t). The horizontal and vertical boundaries of the detector cell are mapped onto the x–z plane (or the y–z plane) and the x–y plane of every voxel slice as the common plane between detector cells and voxels. We consider that the coordinate of the source focal point is $S(x_1, y_1, z_1)$ and that the coordinate of the center of the detector cell is $\det(x_2, y_2, z_2)$. The coordinates of the four boundary points are $(x_2 + d_s\sin\theta/2, y_2 - d_s\cos\theta/2, z_2)$, $(x_2 - d_s\sin\theta/2, y_2 + d_s\cos\theta/2, z_2)$, $(x_2, y_2, z_2 + d_t/2)$, and $(x_2, y_2, z_2 - d_t/2)$. The angle θ is the rotation angle.

Then we need to compute for the slope of the connecting line between the center of the detector cell and the source focal point and decide which plane would be chosen to divide the 3D image. When $|x_2 - x_1| > |y_2 - y_1|$, we divide the 3D image by the y–z plane as shown in Figs. 3(a)–3(d). When $|x_2 - x_1| \le |y_2 - y_1|$, the x–z plane is selected to divide the image. When $|x_2 - x_1| > |y_2 - y_1|$, every detector cell has four beams connected to the focal point, and all beams intersect each y–z plane. We then need to compute the intersection coordinates and indices of the four beams for the i-th y–z plane. The y-coordinates of the intersections of the two beams in the horizontal direction can be calculated as follows:

$$\begin{aligned} y_{i1} &= k_{y1}(i + 1/2 - x_1) + y_1 + N_y/2, \\ y_{i2} &= k_{y2}(i + 1/2 - x_1) + y_1 + N_y/2 \end{aligned} \tag{6}$$

with

$$k_{y1} = \frac{y_2 - \frac{d_s}{2}\cos\theta - y_1}{x_2 + \frac{d_s}{2}\sin\theta - x_1}, \quad k_{y2} = \frac{y_2 + \frac{d_s}{2}\cos\theta - y_1}{x_2 - \frac{d_s}{2}\sin\theta - x_1},$$

and then we obtain the largest integers that are smaller than y_{i1} and y_{i2}, i.e.,

$$Y_{i1} = \lfloor y_{i1} \rfloor, Y_{i2} = \lfloor y_{i2} \rfloor, \tag{7}$$

which are exactly the y-indices of the intersecting voxel candidates. Similarly, we can calculate the z-coordinates of intersections of the two beams in the vertical direction as follows:

$$\begin{aligned} z_{i1} &= \frac{k_{z1}(i + d_s/2 - x_1) + z_1}{d_z} + N_z/2, \\ z_{i2} &= \frac{k_{z2}(i + d_s/2 - x_1) + z_1}{d_z} + N_z/2, \end{aligned} \tag{8}$$

with

$$k_{z1} = \frac{z_2 + d_t/2 - z_1}{x_2 - x_1}, \quad k_{z2} = \frac{z_2 - d_t/2 - z_1}{x_2 - x_1}, \tag{9}$$

and then obtain the largest integers that are smaller than z_{i1} and z_{i2}, i.e.,

$$Z_{i1} = \lfloor z_{i1} \rfloor, Z_{i2} = \lfloor z_{i2} \rfloor, \tag{10}$$

which are exactly the z-indices of the intersecting voxel candidates. We then only deal with 3D forward projection as 2D cases.

The key of this proposed algorithm is to observe that the intersections are classified into only two cases for both the y direction and the z direction because of the fact that the size of the detector cell is usually smaller than or equal to the size of voxel. One is the two intersections in the voxel, and the other is the only intersection in the voxel. Consequently, the four beams of one detector cell intersect with the voxels in each slice of the image with at most four cases (Fig. 3).

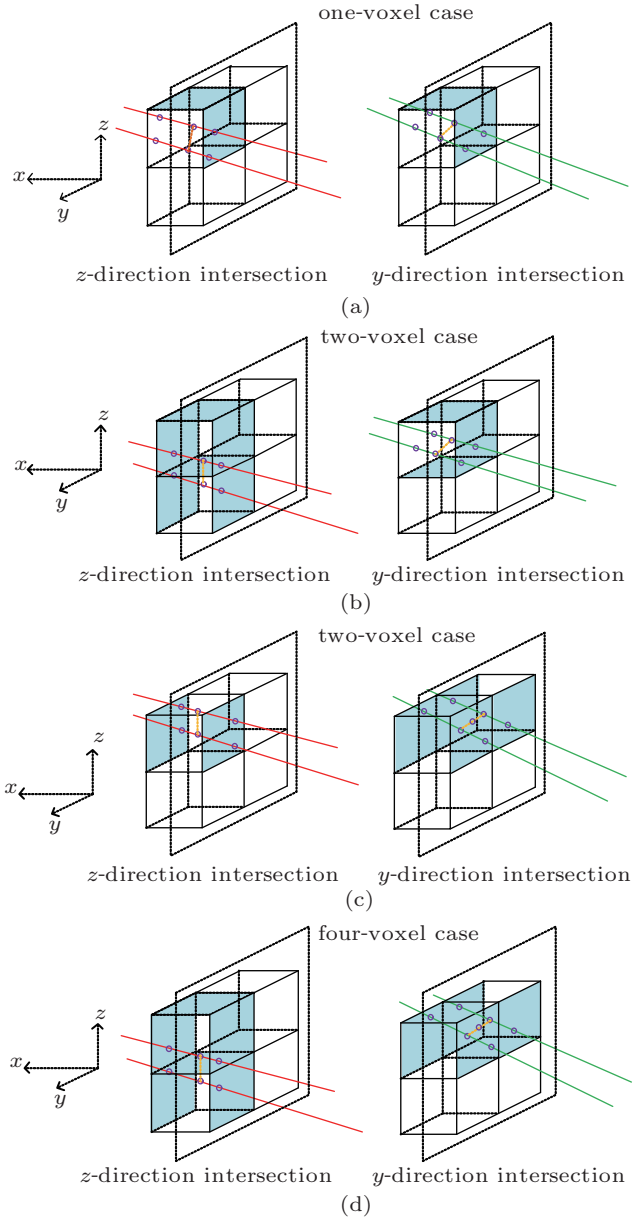


Fig. 3. (color online) Fast algorithm for forward projection operation in 3D cases. When $|x_2-x_1|>|y_2-y_1|$ and $|x_2-x_1|>|z_2-z_1|$, we divide the 3D domain of (N_x,N_y,N_z) voxels into the x planes, and the intersections can be up to three consecutive voxels contained in a 2×2 box; (a) the case with one intersecting voxel along both the y direction (when viewed in the x–y projection plane) and the z direction (when viewed in the x–z projection plane); (b) the case with one intersecting voxel along the y direction and two intersecting voxels along the z direction; (c) the case with two intersecting voxels along the y direction and one intersecting voxel along the z direction; and (d) cases with two intersecting voxels along both y and z directions.

If $Y_{i1}=Y_{i2}$, $Z_{i1}=Z_{i2}$ (Fig. 3(a)), only one voxel in the i-th slice contributes to the detector cell, and the weight can be calculated as follows:

$$\begin{aligned}&(i,Y_{i1},Z_{i1}):l_{t\theta,ijk}=1,\\&\text{if}\quad 1\le Y_{i1}\le N_y,\ \text{ and }\ 1\le Z_{i1}\le N_z.\end{aligned}\tag{11}$$

If $Y_{i1}\neq Y_{i2}$, $Z_{i1}=Z_{i2}$ (Fig. 3(b)), two voxels in the i-th slice in the horizontal direction contribute to the detector cell, and the weights of two intersecting voxels can be calculated as follows:

$$\begin{cases}(i,Y_{i1},Z_{i1}):l_{t\theta,ijk}=\dfrac{\max(Y_{i1},Y_{i2})-y_{i1}}{y_{i2}-y_{i1}},\\ \text{if}\quad 1\le Y_{i1}\le N_y,\ \text{ and }\ 1\le Z_{i1}\le N_z,\\ (i,Y_{i2},Z_{i1}):l_{t\theta,ijk}=\dfrac{y_{i2}-\max(Y_{i1},Y_{i2})}{y_{i2}-y_{i1}},\\ \text{if}\quad 1\le Y_{i2}\le N_y,\ \text{ and }\ 1\le Z_{i1}\le N_z.\end{cases}\tag{12}$$

If $Y_{i1}=Y_{i2}$, $Z_{i1}\neq Z_{i2}$ (Fig. 3(c)), two voxels in the i-th slice in the horizontal direction contributes to the detector cell, and the weights of two intersecting voxels can be calculated as follows:

$$\begin{cases}(i,Y_{i1},Z_{i1}):l_{t\theta,ijk}=\dfrac{\max(Y_{i1},Y_{i2})-y_{i1}}{y_{i2}-y_{i1}},\\ \text{if}\quad 1\le Y_{i1}\le N_y,\ \text{ and }\ 1\le Z_{i1}\le N_z,\\ (i,Y_{i1},Z_{i2}):l_{t\theta,ijk}=\dfrac{z_{i2}-\max(Z_{i1},Z_{i2})}{z_{i2}-z_{i1}},\\ \text{if}\quad 1\le Y_{i1}\le N_y,\ \text{ and }\ 1\le Z_{i2}\le N_z.\end{cases}\tag{13}$$

If $Y_{i1}\neq Y_{i2}$, $Z_{i1}\neq Z_{i2}$ (Fig. 3(d)), four voxels in the i-th slice contribute to the detector cell, and the indices and weights of four intersecting voxels can be calculated as follows:

$$\begin{cases}(i,Y_{i1},Z_{i1}):l_{t\theta,ijk}\\ =\dfrac{\max(Y_{i1},Y_{i2})-y_{i1}}{y_{i2}-y_{i1}}\cdot\dfrac{\max(Z_{i1},Z_{i2})-z_{i1}}{z_{i2}-z_{i1}},\\ \text{if}\quad 1\le Y_{i1}\le N_y,\ \text{ and }\ 1\le Z_{i1}\le N_z,\\ (i,Y_{i2},Z_{i1}):l_{t\theta,ijk}\\ =\dfrac{y_{i2}-\max(Y_{i1},Y_{i2})}{y_{i2}-y_{i1}}\cdot\dfrac{\max(Z_{i1},Z_{i2})-z_{i1}}{z_{i2}-z_{i1}},\\ \text{if}\quad 1\le Y_{i2}\le N_y,\ \text{ and }\ 1\le Z_{i1}\le N_z,\\ (i,Y_{i1},Z_{i2}):l_{t\theta,ijk}\\ =\dfrac{\max(Y_{i1},Y_{i2})-y_{i1}}{y_{i2}-y_{i1}}\cdot\dfrac{\max(Z_{i1},Z_{i2})-z_{i1}}{z_{i2}-z_{i1}},\\ \text{if}\quad 1\le Y_{i1}\le N_y,\ \text{ and }\ 1\le Z_{i2}\le N_z,\\ (i,Y_{i2},Z_{i2}):l_{t\theta,ijk}\\ =\dfrac{y_{i2}-\max(Y_{i1},Y_{i2})}{y_{i2}-y_{i1}}\cdot\dfrac{z_{i2}-\max(Z_{i1},Z_{i2})}{z_{i2}-z_{i1}},\\ \text{if}\quad 1\le Y_{i2}\le N_y,\ \text{ and }\ 1\le Z_{i2}\le N_z.\end{cases}\tag{14}$$

It should be noted that the detector in our text is flat and the weight coefficient calculated should be multiplied by the factors as mentioned in De Man's paper.[21] The factors can be pre-calculated before the intersection weight coefficient, using the in- and out-of-plane angles of the line of interest with the axis perpendicular to the slab of detector bins.

The proposed 3D algorithm is easy to implement for both central processing units (CPUs) and GPUs. For the parallel computing version in the GPUs, we can consider computing the intersections of the beams of one detector cell per parallel thread. Our method for calculating the weight coefficients is less complex than the method for calculating the intersection lengths of beams with each voxel of the 3D image.

2.2.2. Backprojection

The backprojection essence is the adjoint transformation of the forward projection. For parallel computation, we consider that the main loop is the indices of the image voxels and that the weight of each voxel can be calculated per parallel thread. During the implementation, the detector cells that contribute to the voxel in the shadow region must be identified. For the notations, we define (x_a, y_a, z_a) as the coordinate of the center of the voxel, (N_a, N_b) as the number of the detectors along the y and z directions, SO as the distance from the source to the detection line, SD as the distance from the source to the detection line, d_y as the detector cell width, and ϕ as the scan angle.

To simplify the calculation for finding the contributing detector bins, Gao[22] proposed a fast method for calculating the contributing detectors in the shadow region when the voxel and scanning direction are given. In Gao's method, the range of contributing detector bins is extended through a rhombus enclosed in the circumscribed sphere of the voxel to simplify the coordinate calculation.

In our distance-driven backprojection method, the coordinates of the current voxel boundary points B_{left}, B_{bottom}, B_{right}, and B_{top} are acquired after the coordinate system of the reconstructed volume is rotated with the detector being perpendicular to the coordinate axis. As a result, the new coordinates of the boundary points of the voxel are $(x_{B_{\text{left}}}, y_{B_{\text{left}}}, z_{B_{\text{left}}})$, $(x_{B_{\text{right}}}, y_{B_{\text{right}}}, z_{B_{\text{right}}})$, $(x_{B_{\text{top}}}, y_{B_{\text{top}}}, z_{B_{\text{top}}})$, $(x_{B_{\text{bottom}}}, y_{B_{\text{bottom}}}, z_{B_{\text{bottom}}})$ expressed as Eq. (15)

$$
\begin{aligned}
x_{B_{\text{left}}} &= \left(x_a - \frac{1}{2}\right)\cos\phi + \left(y_a - \frac{1}{2}\right)\sin\phi,\\
y_{B_{\text{left}}} &= -\left(x_a - \frac{1}{2}\right)\sin\phi + \left(y_a - \frac{1}{2}\right)\cos\phi,\\
x_{B_{\text{right}}} &= \left(x_a + \frac{1}{2}\right)\cos\phi + \left(y_a + \frac{1}{2}\right)\sin\phi,\\
y_{B_{\text{right}}} &= -\left(x_a + \frac{1}{2}\right)\sin\phi + \left(y_a + \frac{1}{2}\right)\cos\phi,\\
z_{B_{\text{left}}} &= z_{B_{\text{right}}} = z_a\\
x_{B_{\text{top}}} &= x_{B_{\text{bottom}}} = x_a\cos\phi + y_a\sin\phi,\\
y_{B_{\text{top}}} &= y_{B_{\text{bottom}}} = -x_a\sin\phi + y_a\cos\phi,\\
z_{B_{\text{top}}} &= z_a + \frac{1}{2}, \quad z_{B_{\text{bottom}}} = z_a - \frac{1}{2}.
\end{aligned}
\tag{15}
$$

Four varying variables b_{l}, b_{b}, b_{r}, and b_{t} that are projective coordinates of the current voxel project on the detector are shown in Fig. 4 and the coordinates are expressed as Eq. (15). Consequently, we can find the detector bound along the y- and z-directions through

$$
\begin{cases}
b_{\text{l}} = \left\lfloor [y_{B_{\text{left}}} \cdot SD/(x_{B_{\text{left}}} + SO)]/d_y + N_a/2 \right\rfloor,\\
b_{\text{r}} = \left\lfloor [y_{B_{\text{right}}} \cdot SD/(x_{B_{\text{right}}} + SO)]/d_y + N_a/2 \right\rfloor,
\end{cases}
\begin{cases}
b_{\text{t}} = \left\lfloor [z_{B_{\text{top}}} \cdot SD/(x_{B_{\text{top}}} + SO)]/d_z + N_b/2 \right\rfloor,\\
b_{\text{b}} = \left\lfloor [z_{B_{\text{bottom}}} \cdot SD/(x_{B_{\text{bottom}}} + SO)]/d_z + N_b/2 \right\rfloor.
\end{cases}
\tag{16}
$$

After the contributing bins are acquired, which are often only a few, the backprojection weight coefficient of the given voxel with the contributing bins can be easily computed. Although the calculation is more complex in the backprojection to match the system matrix of projection, the algorithm can also achieve a good performance for the parallel computation.

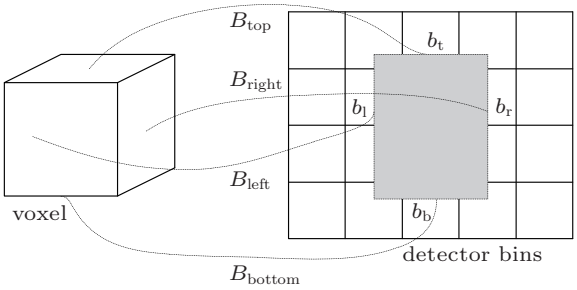


Fig. 4. Four projective mappings B_{left}, B_{bottom}, B_{right}, and B_{top} of the coordinates of the current voxel project on the detector corresponding to four varying variables b_{l}, b_{b}, b_{r}, and b_{t}.

3. Experimental results

3.1. Simulation studies

To evaluate the performance of the Compute Unified Device Architecture (CUDA)-aided implementation, we implement and test the operation of forward projection and backprojections on both CPU and GPU. To verify our forward projection and backprojection implementations, we concentrate on the SART algorithm[23] that is classical in the algebraic reconstruction algorithms[21,23,24] to test the GPU-based DDM with simulated and physical experiments. Furthermore, two data sets are used for NVIDIA GPU architecture Tesla K20c. This GPU device has 2496 CUDA cores and 5120 MB global memory. In the performance test, a 3D Shepp-Logan phantom with an attenuation coefficient of 0.0 to 1.0 is utilized. A single circle trajectory is utilized for cone beam scanning. The source to axis distance is 30 cm, and the distance of the source to the center of the flat panel is 60 cm. The detector panel has a size of 12.8 cm × 12.8 cm, and the phantom has a size of 6.4 cm × 6.4 cm × 6.4 cm. The projection data are collected by 360 equally angular views in 360°.

Two types of discretization are applied to test the two operations under different data sets (Table 1). All experiments are performed on the workstation configured with dual cores Intel Xeon CPU of E5-2620 @ 2.10 GHz (only one core is used) and a NVIDIA GPU card of Tesla K20c. The time consumption for both CPU and different GPUs is listed in Table 2 together with the speedup. The time consumption is calculated through the statistical average of 50 iterations. The speedup in Table 2 shows that the acceleration strategy applied in this study can considerably improve the performance with GPU.

Table 1. Dataset for situations 1 and 2.

	Volume data	Projection data	Voxel size/mm	Detector bin size/mm
Situation 1	128×128 × 128	256 × 256 × 360	0.5	0.5
Situation 2	256 × 256 × 256	512 × 512 × 360	0.25	0.25

Table 2. Running time for related operation in reconstruction (unit for time: seconds).

	Situation 1			Situation 2		
	CPU/s	GPU/s	Speedup/s	CPU/s	GPU/s	Speedup/s
Forward projection	103	0.59	174.5	834	4.5	185.3
Backprojection	281	1.4	201.1	2223	10.5	211.8

To verify the reconstruction accuracy for both the CPU-based and GPU-based methods, we calculate the root-mean-square error (RMSE) of each reconstruction, which is obtained by

$$\mathrm{RMSE} = \sqrt{\sum \frac{(x_{\mathrm{CPU}} - x_{\mathrm{GPU}})^2}{N}}. \tag{17}$$

Table 3. RMSE of GPU computation for related operation.

	Forward projection/10^{-6}	Backprojection/10^{-6}
Situation 1	2.3	1.7
Situation 2	1.5	0.9

In the simulation study, we only test the performance of our model with computing efficiency in the forward and back-projection, and in the next section we will test the performance of the model with both computing efficiency and imaging quality.

3.2. Real data reconstruction verifications

The reconstruction algorithm proposed in this study is composed of projection and backprojection, with a few vector additions and comparisons. In this sub-section, reconstruction is performed using both simulation data and real CT projections. The goal is to test the performance of the new model and the quality of image reconstruction. For real data reconstruction, the projections of a medical head phantom are acquired with the CBCT system, which mainly consists of a flat-panel detector (Varian4030E, USA) and an x-ray source (Hawkeye 130, Thales, France). Table 4 lists the scanning and reconstruction parameters. The result also proves that almost no difference exists between the RMSEs of the CPU-based and GPU-based reconstructions.

We perform an intersecting-line model and a DDM in iterative reconstruction with real data with SART. Although the system matrix model is significantly accurate with the intersecting-line length model, the ray-driven result also shows the interpolation artifacts (Fig. 4). The artifacts can be completely eliminated using DDM (Fig. 4). Compared with the ray-driven model in Gao's method,[22] our GPU-based DDM maintains the distance-model characteristics to prevent the high-frequency artifacts caused by the discretization of detector cells. As shown in Table 5, the experiment for real data verifies our proposed method for improving the computational efficiency and achieving a high acceleration ratio.

Table 4. Parameters in a CBCT scan.

Parameters	
Source to axis distance	678 mm
Source to detector distance	1610
Views of projection data	360 views
Projection size	768 pixels × 432 pixels
Detector cell size	0.508 mm × 0.508 mm
Reconstruction size	384 voxels × 384 voxels × 216 voxels
Voxel size	0.214 mm × 0.214 mm × 0.214 mm

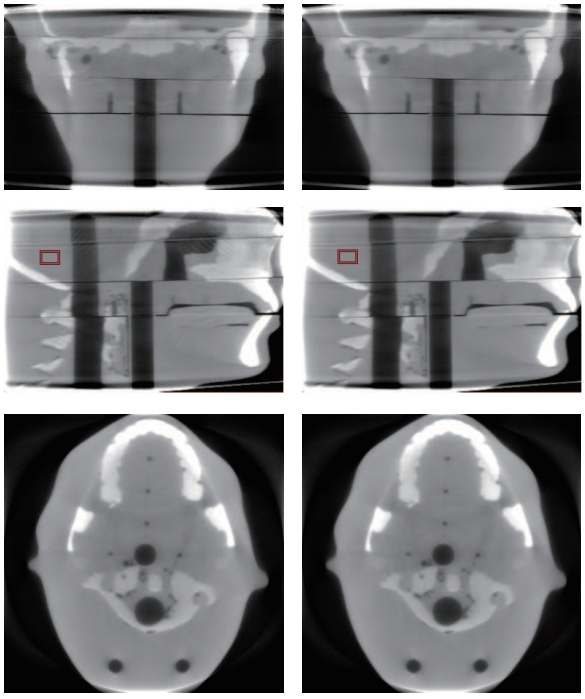


Fig. 5. (color online) Reconstructions of real CT data experiments. First and second columns from left to right are results of (a) Gao's method and (b) the distance-driven model, both with GPU acceleration. Images from top to bottom row show the results of median sagittal, central coronal, and central transverse sections. The red rectangle shows ROI used for SNR calculation.

Table 5. Running time for real data experiments of the SART algorithm with two models.

	CPU performance/s	GPU performance/s	Acceleration ratio
Gao's method	770448	4586	167.9
This study	482944	2816	171.5

To quantify the image quality of reconstructed images in terms of image noise, the signal to noise ratio (SNR) of a region of interest (ROI) is shown in Table 6. The calculation is as follows:

$$\mathrm{SNR} = 10 \times \log_{10}\left[\frac{E^2(p)}{E([p-E(p)]^2)}\right], \tag{18}$$

where p is the set of pixels inside the ROI and stands for a spatial average over the ROI. It can be observed from Table 6 that the SNR of images reconstructed by our method is higher than Gao's method and it could indicate our method has the advantage of suppressing the noise.

Table 6. SNR of the ROI for Gao's method and this method.

	Gao's method	This method
SNR	32.1	37.8

We perform additional reconstructions using the SART algorithm with two projector–backprojector pairs to compare the performance of our proposed methods with other methods. The experiments are prepared with the pair of ray-driven and voxel-driven backprojectors as well as the unmatched pair of distance-driven and backprojectors. The unmatched pair with ray-driven and voxel-driven backprojectors is generally used in the conventional GPU-based method. In this method, we consider the unmatched DD-PRJ/DD-BPR pair used in Daniel's study,[25] in which the backprojection operation is approximated to accelerate the computational speed. Figure 6 shows the performance of the three methods in real data reconstruction. This figure also illustrates a noticeable artifact caused by the unmatched projector/backprojector pair. The unmatched model accelerates the reconstruction computational efficiency but causes inaccurate results. In addition, this model affects the analysis and evaluation of the reconstructed real image. Table 7 shows the SNR of three methods in the performance of imaging reconstruction. It also can prove that our method has the better performance quality in the reconstruction.

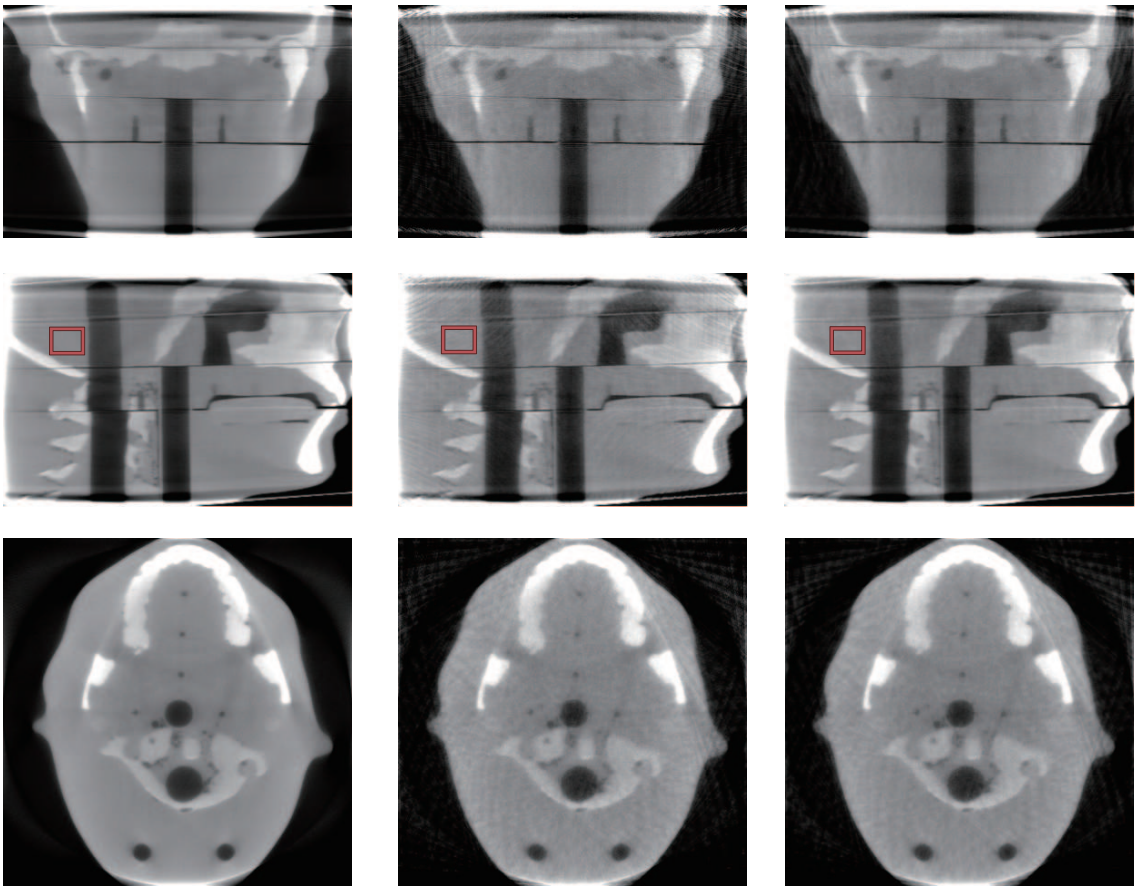


Fig. 6. (color online) Three contrast reconstructions of real CT data experiments. First to third columns from left to right are results of (a) matched distance-driven model and (b) ray-driven projection/voxel-driven backprojection model, and (c) unmatched distance-driven model with GPU acceleration. Images from top to bottom row show the results of median sagittal, central coronal, and central transverse sections. The rectangle shows ROI used for SNR calculation.

Table 7. SNR of the ROI for three contrast methods.

	This method	RDM-VDM (GPU)	Unmatched DD (GPU)
SNR	37.8	29.4	32.1

4. Summary and conclusion

We developed a fast parallel algorithm for the 3D distance-driven model which is known as an accurate and low-computational complexity projection and backprojection model. Our algorithm results in a matched projector–backprojector frame through the highly effective parallelization of both forward and backprojections. The key to our efficient algorithm is to exploit structure in the pattern of the cone beam rays, and decompose the problem into components in the slices of the 3D volume.

The simulation and real data experimental results show that the fast parallel GPU-based algorithm for the distance-driven model with respect to the optimized CPU-based method significantly accelerates the computational efficiency in iterative reconstruction. As expected, the reconstructed images using our GPU-based methods are identical to those using the conventional CPU-based methods. Compared with the other 3D projection and backprojection computational methods, such as the fast interacting-line length model, our proposed method has a better performance for balancing computational complexity and accuracy. In contrast to conventional GPU-accelerated models in which the reconstruction accuracy is often sacrificed for high-performance computation, our GPU-accelerated model can be guaranteed to maintain the accuracy with an accelerated system matrix model under a matched projector/backprojector frame.

In this study, we proposed a parallelizable algorithm for computing forward and backprojections. In the future, more optimization methods for CUDA or OpenCL can be used to optimize the parallel computation. The performance of the method can be tested in more iterative reconstruction algorithms, such as total variation regularization methods.

References

[1] Yang F Q, Zhang D H, Huang K D, Wang K and Xu Z 2014 *Acta Phys. Sin.* **63** 058701 (in Chinese)

[2] Wang L Y, Zhang H M, Cai A L, Yan B, Li L and Hu G E 2013 *Acta Phys. Sin.* **62** 198701 (in Chinese)

[3] Bian J, Siewerdsen J, Han X, Sidky E Y, Prince J L, Pelizzari C A and Pan X C 2010 *Phys. Med. Biol.* **55** 6575

[4] Zhang H M, Wang L Y, Yan B, Li L, Xi X Q and Lu L Z 2013 *Chin. Phys. B* **22** 078701

[5] Wang L Y, Li L, Yan B, Jiang C S, Wang H Y and Bao S L 2010 *Chin. Phys. B* **19** 088106

[6] Jia X and Lou Y F 2011 *Phys. Med. Biol.* **47** 77

[7] De Man B and Fessler J A 2010 *Biomedical Mathematics: Promising Directions in Imaging, Therapy Planning, and Inverse Problems* (Medical Physics Publishing) pp. 113–140

[8] Siddon R L 1985 *Med. Phys.* **12** 252

[9] Joseph P M 1983 *IEEE Trans. Med. Imaging* **1** 192

[10] Wang G, Vannier M W and Cheng P C 1999 *Microsc. Microanal.* **5** 58

[11] De Man B and Basu S 2002 *IEEE Nuclear Science Symposium Medical Imaging Conference*, Norfolk, IEEE, p. 1477

[12] Lewitt R M 1990 *J. Opt. Soc. Am.* **7** 1834

[13] Yu H and Wang G 2012 *J. X-Ray Sci. Technol.* **20** 229

[14] Zhang O, Connor Y and Fessler J A 2006 *IEEE Trans. Med. Imaging* **25** 582

[15] Vazquez F, Garzon E M and Fernandez 2011 *The Comput. J.* **54** 1861

[16] Herman G T 1980 *Image Reconstruction from Projections: the Fundamentals of Computerized Tomography* (New York: Academic Press) p. 20

[17] Zeng G L and Gullberg G T 2000 *IEEE Trans. Med. Imaging* **19** 548

[18] Nguyen V G and Lee S J 2012 *Opt. Eng.* **51** 093203

[19] Long Y, Fessler J A and Balter J M 2010 *IEEE Trans. Med. Imaging* **29** 1839

[20] De Man B and Basu S 2004 *Phys. Med. Biol.* **49** 2463

[21] Andersen A H and Kak A C 1984 *Ulwason. Imag.* **6** 81

[22] Gao H 2012 *Med. Phys.* **39** 7110

[23] Gordon R, Bender R and Herman G T 1970 *J. Theor. Biol.* **29** 471

[24] Wang G and Jiang M 2004 *J. X-Ray Sci. Technol.* **12** 169

[25] Daniel G, Ulrich H, Ralf S, Elmar S and Ulrich S 2009 *SIAM J. Imag. Sci.* **31** 4204